horticulturae

MDPI

Article

Study on Target Detection Method of Walnuts during Oil Conversion Period

Xiahui Fu [1], Juxia Wang [1,*], Fengzi Zhang [1], Weizheng Pan [1], Yu Zhang [1] and Fu Zhao [1]

College of Agricultural Engineering, Shanxi Agricultural University, Jinzhong 030801, China; yuzhang976843@163.com (Y.Z.)
* Correspondence: juxiawang11@163.com

Abstract: The colors of walnut fruits and leaves are similar in the oil transformation period, and the fruits are easily blocked by the branches and leaves. On the basis of the improved YOLOv7-tiny, a detection model is proposed and integrated into an Android application to solve the problem of walnut identification. Ablation experiments conducted with three improved strategies show that the strategies can effectively enhance the performance of the model. In terms of combinatorial optimization, the YOLOv7-tiny detection model that combines FasterNet and LightMLP modules works excellently. Its AP_{50} and $AP_{50–95}$ are 3.1 and 4 percentage points (97.4% and 77.3%, respectively) higher than those of the original model. YOLOv7-tiny's model size and number of parameters are reduced by 14.6% and 14.4%, respectively, relative to those of the original model, and its detection time decreases to 15.4 ms. The model has good robustness and generalization ability and can provide a technical reference for intelligent real-time detection of walnuts during the oil conversion period.

Keywords: walnuts; oil conversion period; target detection; ablation test

check for updates

Citation: Fu, X.; Wang, J.; Zhang, F.; Pan, W.; Zhang, Y.; Zhao, F. Study on Target Detection Method of Walnuts during Oil Conversion Period. *Horticulturae* **2024**, *10*, 275. https://doi.org/10.3390/horticulturae10030275

Academic Editor: Darius Kviklys

Received: 31 January 2024
Revised: 3 March 2024
Accepted: 8 March 2024
Published: 12 March 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Walnuts are rich in various micronutrients and unsaturated fatty acids needed by the human body; they can reduce the possibility of depression, Alzheimer's disease, and cancer and are one of the most popular nuts [1]. Shanxi is a large walnut cultivation province, and in recent years, the green walnut has been increasingly adopted in food and beverage development because of its unique taste and has gradually entered the market and become popular [2]. Target detection of green walnuts is essential for automatically picking fresh fruits and alleviating labor shortage in orchards. The oil transformation period of walnuts, which is the growth stage before maturity, is from early July to the end of August. During this time, the branches and leaves of walnut trees are thriving in the natural environment, and the color of the green walnut's fruits is similar to that of its leaves. At the same time, the branches, leaves, and fruits exhibit varying degrees of shading. As a result, the difficulty of green walnut recognition increases [3].

With the continuous development of intelligent agriculture, target detection technology has been introduced to agricultural product detection and resulted in the gradual replacement of manual identification with recognition and feature extraction methods, such as histogram of oriented gradients and scale-invariant feature transform. Common target detection models include the two-stage region-based convolutional neural network (R-CNN) [4], Fast R-CNN [5], and Faster R-CNN [6]. Single-stage target detection models include the single-shot multibox detector (SSD) [7] and the You Only Look Once (YOLO) series [8–11].

Extensive research on the detection of targets with similar colors of fruits and leaves has been conducted at home and abroad. For the detection of green walnuts, Fan et al. used the pretrained VGG16 network structure as the feature extractor of the Faster R-CNN, introduced batch normalization in the convolutional layer, and improved the region proposal network structure via bilinear interpolation to increase the precision rate, recall rate, and F1 score while accelerating image detection relative to the original Faster R-CNN [12].

Hao et al. replaced the backbone network DarkNet-53 in YOLOv3 with MobilNet-v3, imported the Mixup data enhancement method, and adjusted the anchor frames in accordance with the labeled frame size statistics, thus increasing the accuracy rate to 94.52% [13]. For young apple fruit detection, Tian et al. used DenseNet instead of the low-resolution transport layer in the YOLOv3 model, and the training results showed that the modified YOLOv3 model outperforms the original YOLOv3 model and Faster R-CNN that uses the VGG16 network; the F1 score for the young fruit type is 83.2% [14]. Song et al. inserted an efficient channel attention mechanism into the three reparameterized paths of the YOLOv7 model and found that the modified YOLOv7 model improves the evaluation indicators of the Faster R-CNN, SSD, scaled YOLOv4, YOLOv5, YOLOv6, and YOLOv7 in the case of noise blur, shadow, and severe occlusion [15]. For the detection of low-quality young fruits, the nonlocal attention module and the convolutional block attention module were fused with the YOLOv4 model in a previous study [16], and the average accuracy of images with highlights, shadows, and blur increased. For green pepper detection, Chen et al. proposed an improved popular ranking significant object detection algorithm with an accuracy of 85.6% [17]. For green apple detection, Bargoti and Underwood developed a Faster R-CNN fruit detection system that uses ImageNet pretrained models and obtained an F1 score of over 90% for green apple image detection [18].

Currently, target detection technology is mostly used to detect apples [19–21], tomatoes [22–24], dragon fruits [25–27], and citruses [28–30], whose fruits have a bright color, large size, and easily recognizable characteristics. The identification of green walnuts is difficult because of their small size and the similar colors of the fruits and leaves. Fan et al. realized the detection of green walnuts by improving the Faster R-CNN. However, the detection time of a single image was 227 ms, which was long and affected the real-time detection performance; moreover, the specific size of the model was not given [12]. Hao et al. used the MobileNet-v3 network to optimize YOLOv3, and the detection time and detection speed of a single image reached 31 frames/s. However, the optimized YOLOv3 model's size is 88.6 MB, which is still large [13]. The difference between YOLOv7 and YOLOv5 is the design of ELAN in the former to solve the problem of gradual deterioration of the convergence of deep models during model scaling. YOLOv7-tiny is a variant of YOLOv7. Compared with YOLOv7, YOLOV7-tiny targets lightweight target detection tasks and has a smaller model volume and faster inference; it is suitable for scenarios where target detection tasks are not highly complex.

Given this background, a target detection algorithm based on improved YOLOv7-tiny was proposed in this study. The proposed algorithm incorporates the FasterNet module into the backbone network to reduce the number of model parameters and improve the detection efficiency; adds the LightMLP module to the head structure to improve the model's image detection effect in complex environments, such as backlight and severe shade; and introduces the lightweight upsampling operator content-aware reassembly of features (CARAFE) to optimize the image resolution of the extracted feature map after magnification. Rapid, high-precision detection of green walnuts was realized.

2. Materials and Methods

2.1. Image Acquisition Preprocessing and Annotation

Experimental data were collected from Shanxi Agricultural University and the Fruit Tree Research Institute of Shanxi Academy of Agricultural Sciences. An iPhone was used for image acquisition, and the acquisition period was early August and divided into morning, midday, and afternoon shooting. The image resolution was 3024 × 3024 pixels in JPG format. In consideration of the diversity of natural environments, the walnut images were collected in various conditions, including smooth light, backlight, slight shade, and severe shade. The four types of collected pictures are shown in Figure 1.

The dataset was expanded using various preprocessing methods to prevent overfitting during model training. Random rotation can lead to the loss or mutilation of target objects in some images, so random rotation of batch images was not performed in this study. The

initially acquired images were uniformly reduced in saturation, increased in saturation and flipped vertically, reduced in brightness and flipped vertically, and added with Gaussian noise and flipped horizontally to expand the dataset. The resulting dataset had 10,125 images and a size of 13.7 GB. The randomly sampled images were divided into training, validation, and test sets at a ratio of 8:1:1. The preprocessing operations are shown in Figure 2.



(**a**) (**b**) (**c**) (**d**)

Figure 1. Image preprocessing example. (**a**) Smooth light; (**b**) backlight; (**c**) slight shade; (**d**) severe shade.



(**a**) (**b**) (**c**) (**d**) (**e**)

Figure 2. Image preprocessing example. (**a**) Original image; (**b**) desaturation; (**c**) increase saturation and vertical flip; (**d**) reduce brightness and vertical flip; (**e**) add Gaussian noise and horizontal flip.

As indicated in Figure 3, after the image size was uniformly adjusted to 640 × 640 pixels, LabelImg software v1.8.1. was employed to annotate the divided dataset. In Figure 3, four green walnuts are marked by purple rectangular boxes labeled "tender walnuts", and a corresponding .txt file was generated for each image to store the coordinate data of the marked target.

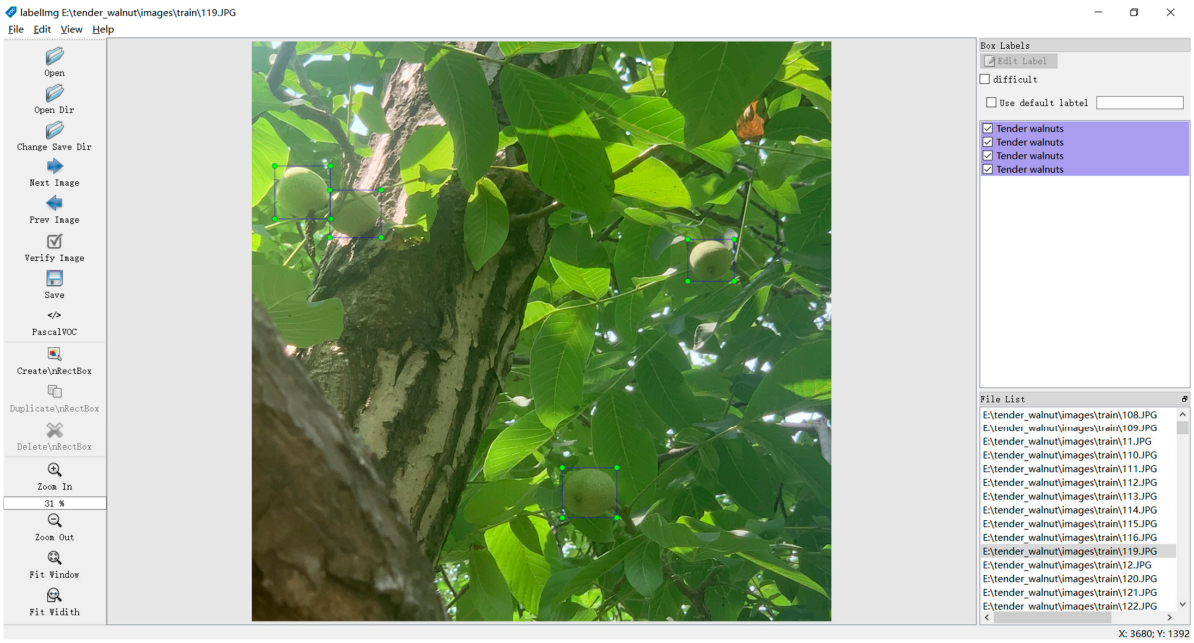


Figure 3. LabelImg software was used to mark green walnuts.

2.2. FasterNet

Partial convolution (PConv) and the FasterNet modules are shown in Figure 4. The residual idea was adopted as a whole in the FasterNet modules, and each FasterNet module consisted of one PConv layer followed by two 1 × 1 regular convolution layers [31]. The batch normalization operation and the activation function rectified linear unit (ReLU) were placed behind the intermediate convolution to improve neural network performance.

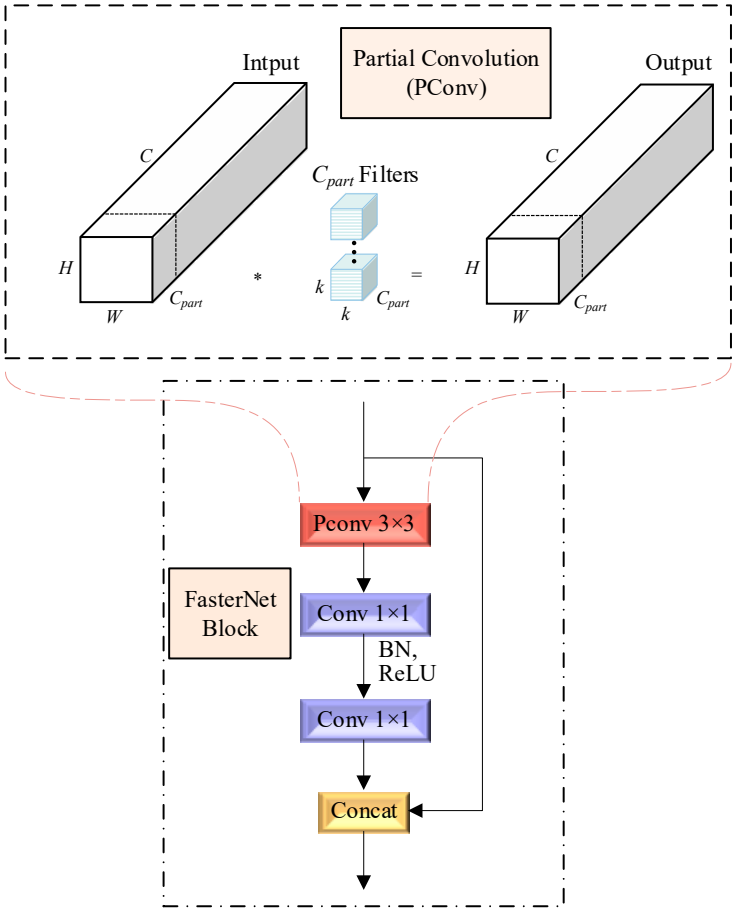


Figure 4. Partial convolution and FasterNet module.

In cases where the input and output feature mappings have the same number of channels, PConv reduces computational redundancy and memory accesses by applying regular convolution only on some of the input channels for spatial feature extraction and keeping the rest of the channels unchanged. For consecutive or regular memory accesses, the first or last consecutive partial channel is considered a representation of the whole feature mapping used for computation. When the number of partial channels is one-quarter of the overall number of channels, the number of floating point operations (FLOPs) and memory accesses for PConv is only one-quarter of that for regular convolution and is calculated as follows:

$$F = H \times W \times {C_{part}}^2 \quad (1)$$

$$M = H \times W \times 2C_{part} + k^2 \times {C_{part}}^2 \approx H \times W \times 2C_{part} \quad (2)$$

where F is the number of FLOPs, M represents memory access, H is the height of the feature map, W is the width of the feature map, C_{part} is the number of partial channels, and k refers to the width and height of the feature map obtained by partial convolution after filtering.

2.3. Lightweight Upsampling Operator CARAFE

The lightweight upsampling operator CARAFE has a large sensory field during reassembly to maximize the use of the surrounding information and can guide the reassembly process on the basis of the semantic information of the input feature graph while using only a few parameters and minimal computational effort [32]. The specific structure of the CARAFE upsampling module is shown in Figure 5. CARAFE is mainly divided into upsampling kernel prediction and feature reorganization modules. As indicated in Figure 5, the kernel prediction module has three operations. First, given an input feature map with shape $H \times W \times C$, the number of channels is compressed into C_m by using 1×1 convolution. Second, the upsampling magnification is set to 2. Assuming that the upsampling core size is $k_r \times k_r$, the feature map is reorganized into $2H \times 2W \times k_r \times k_r$ through pixel shuffling [33]. Last, normalization is performed by softmax. The task of the feature recombination module is to pass the kernel prediction module to the upsampling kernel that is the recombination module and the feature area of the input feature map and then implement the dot product between the upsampling kernel and the feature area; afterward, an output feature map with a shape of $2H \times 2W \times C$ is obtained.

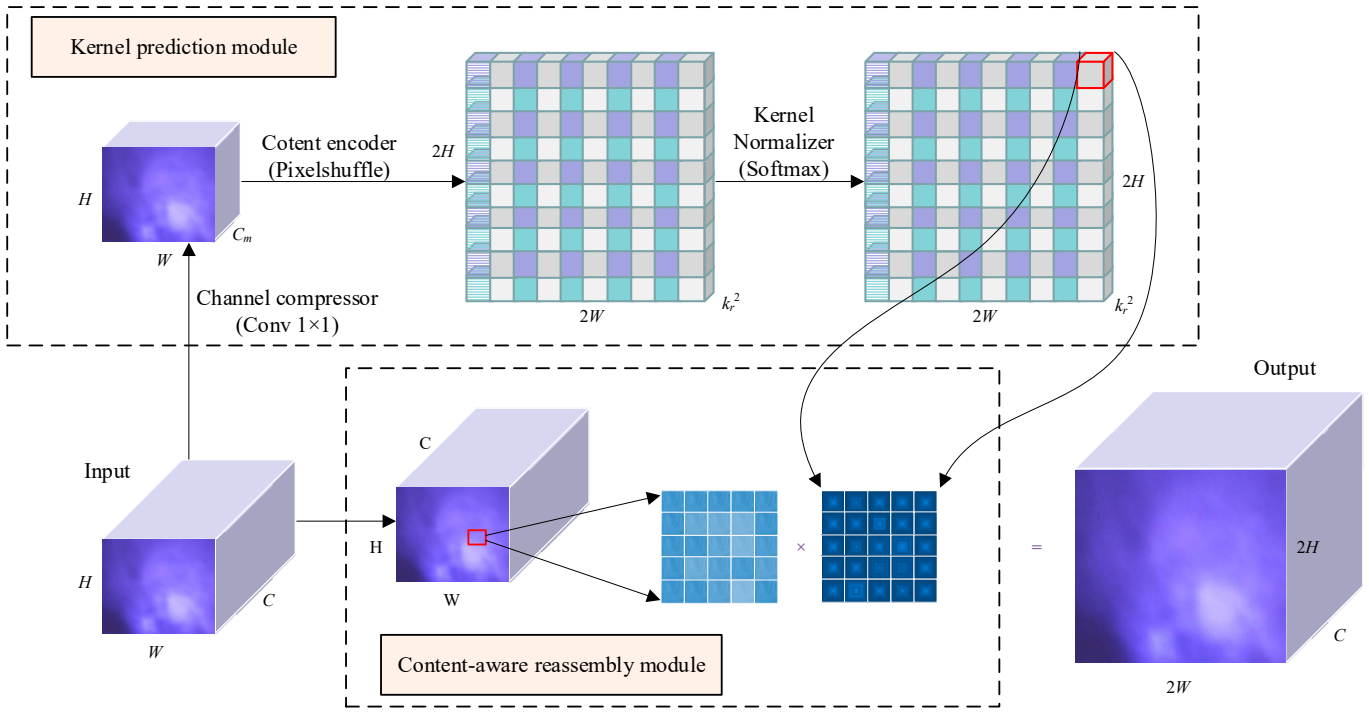


Figure 5. CARAFE upsampling.

2.4. LightMLP

Multilayer perception (MLP), also known as the multilayer fully connected network, consists of an input layer, one or more hidden layers, and an output layer. In this study, two lightweight LightMLP modules [34] were fused into the YOLOv7-tiny header structure and then placed behind two upsampling blocks of the header structure to facilitate the capture of global features and consequently improve the model's performance. The LightMLP module structure is shown in Figure 6, where GN denotes group normalization, DConv refers to depth-wise convolution, the channel scaling [35] function is for channel scaling, DropPath [36] is a network branch that acts as a regularization method, and channel MLP [37] is a multilayer perceptron based on the channel.

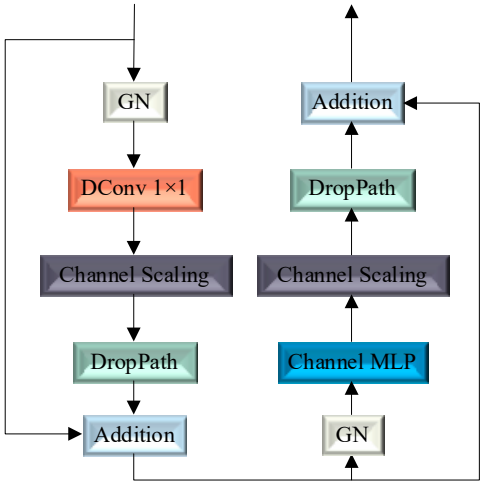


Figure 6. LightMLP block.

2.5. Improved YOLOv7-Tiny Network Structure

YOLOv7-tiny has a smaller footprint and better real-time detection performance compared with the YOLOv7 model, and it is adaptable to embedded devices and suitable for research under limited computational resources. The overall method of improving the YOLOv7-tiny structure is shown in Figure 7. The specific improvement strategies are as follows: fusing the FasterNet module into the ELAN layer of the backbone network and introducing the Pconv and CBR modules; inputting the CARAFE lightweight upsampling operator instead of the nearest-neighbor interpolation upsampling of the original network; and inserting the lightweight MLP module after the two upsampling blocks of the head structure. Figure 8 shows the .yaml file where FasterNet, CARAFE, and LightMLP modules are integrated into the YOLOv7-tiny structure. Lines 45–50 in Figure 8 correspond to the ELAN–FasterNet module in Figure 7, line 66 corresponds to CARAFE in gray in Figure 7, and line 68 corresponds to LightMLP in red in Figure 7. In this study, ablation tests were performed to verify the effectiveness of each optimization strategy in single and combined optimization. The MP module was for downsampling, the three SP modules for the maximum pooling operation were used to obtain different sensory fields, and the nearest module represented the nearest-neighbor interpolation upsampling of the original YOLOv7-tiny.

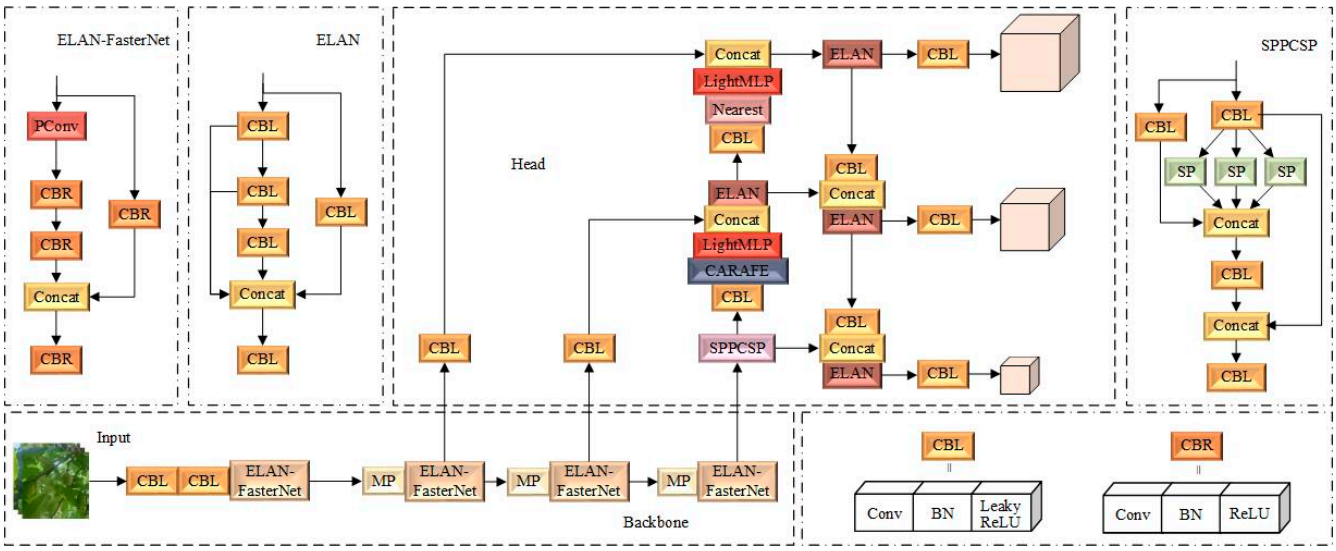


Figure 7. The improved YOLOv7-tiny network structure.

```
   [-1, 1, PConv, [256]],
   [-2, 1, Conv, [256, 1, 1, None, 1, nn.ReLU()]],
   [-1, 1, Conv, [256, 3, 1, None, 1, nn.ReLU()]],
   [-1, 1, Conv, [256, 3, 1, None, 1, nn.ReLU()]],
   [[-1, -3], 1, Concat, [1]],
   [-1, 1, Conv, [512, 1, 1, None, 1, nn.ReLU()]],
  ]

# yolov7-tiny head
head:
  [[-1, 1, Conv, [256, 1, 1, None, 1, nn.LeakyReLU(0.1)]],
   [-2, 1, Conv, [256, 1, 1, None, 1, nn.LeakyReLU(0.1)]],
   [-1, 1, SP, [5]],
   [-2, 1, SP, [9]],
   [-3, 1, SP, [13]],
   [[-1, -2, -3, -4], 1, Concat, [1]],
   [-1, 1, Conv, [256, 1, 1, None, 1, nn.LeakyReLU(0.1)]],
   [[-1, -7], 1, Concat, [1]],
   [-1, 1, Conv, [256, 1, 1, None, 1, nn.LeakyReLU(0.1)]],

   [-1, 1, Conv, [128, 1, 1, None, 1, nn.LeakyReLU(0.1)]],
   [-1, 1, CARAFE, [3, 5]],
   [21, 1, Conv, [128, 1, 1, None, 1, nn.LeakyReLU(0.1)]],
   [-1, 1, LightMLPBlock, []],
   [[-1, -2], 1, Concat, [1]],
```

Figure 8. The core code of improved YOLOv7-tiny network structure. The three red boxes in the figure represent ELAN–FasterNet, CARAFE, and LightMLP from top to bottom.

2.6. Evaluation Index

Average precision (AP), model size (M_s), number of parameters (P_a), and detection time (T) were selected as model evaluation indexes.

AP is the area enclosed by the coordinate axis and the P-R curve, which is composed of precision (P) and recall (R). A large area indicates the good overall performance of the model. P is the proportion of actual positive samples to all predicted positive samples, and R is the proportion of positive samples that are correctly recognized based on the true class. The operational formula is as follows:

$$P = \frac{T_p}{T_p + F_p} \times 100\% \tag{3}$$

$$R = \frac{T_p}{T_p + F_n} \times 100\% \tag{4}$$

$$AP = \int_0^1 P(R)dR \tag{5}$$

where T_p and F_p represent true positive and false positive, respectively, and T_n and F_n refer to true negative and false negative, respectively.

Two specific average accuracy metrics, namely, AP_{50} and $AP_{50–95}$, were used in this study. AP_{50} is the average accuracy when the intersection over union (IoU) threshold is 50%. $AP_{50–95}$ is the average AP of 10 different IoU thresholds from 50% to 95% increasing in steps of 5%.

Model size is the size of the memory space occupied by the model, and the number of parameters reflects model complexity. Detection time is the average time used by the model to detect a single image.

2.7. Test Environment and Hyperparameter Setting

The computer used for the experiment had Windows 10 Professional OS, the CPU was Intel Core i7-7700HQ operating at 2.80 GHz, and the operating memory was 16 GB. The system type was a 64-bit operating system with an x64-based processor and an NVIDIA GeForce GTX 1050 Ti graphics card with a video memory of 4 GB. The PyTorch deep learning framework, Python 3.8.3, Torchvision 0.9.2, and CUDA 11.6 were used for model training.

The training number was 300, the batch size was 16, the parameter optimizer was stochastic gradient descent, the momentum was 0.937, the weight decay was 0.0005, and the initial learning rate was 0.01.

3. Results

3.1. Training and Testing Results of Three Separate Improved Methods

Figure 9 shows the performance of the three improved strategies in the four parameter indicators of precision, recall, AP_{50}, and $AP_{50–95}$ in the training process. Among the strategies, YOLOv7-tiny + FasterNet had the best performance (ranked first) in all four parameters, YOLOv7-tiny + CARAFE ranked last, and YOLOv7-tiny + LightMLP was in the middle.

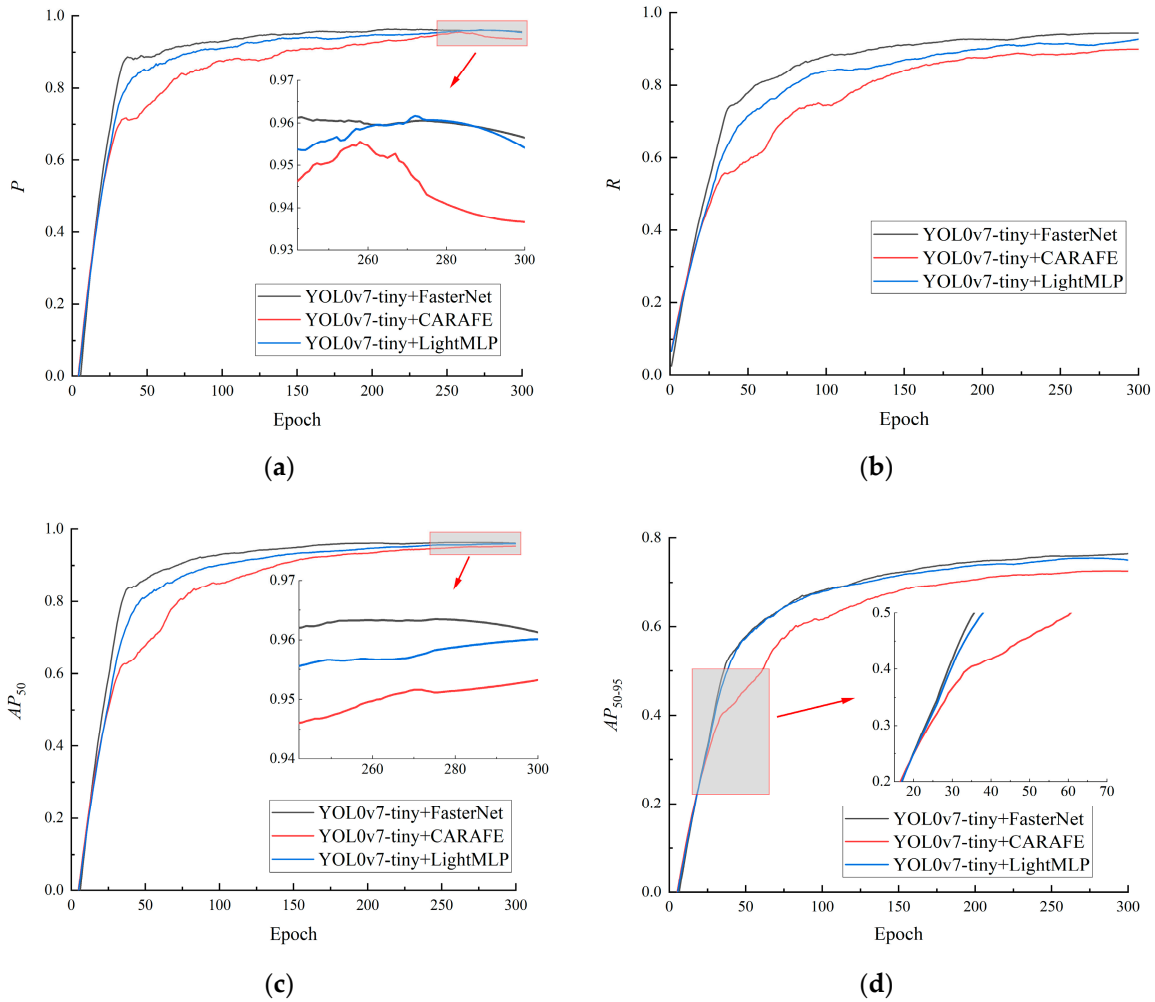


Figure 9. The change curves of each parameter index in the training process of three improved schemes. (**a**) Precision; (**b**) recall; (**c**) AP_{50}; (**d**) $AP_{50–95}$.

The performance of the three improved schemes on the test set is shown in Table 1. Among the three schemes, YOLOv7-tiny + LightMLP had the highest precision of 97.0%, YOLOv7-tiny + CARAFE was in the middle in terms of model size and number of pa-

rameters, and YOLOv7-tiny + FasterNet had the lowest performance in terms of model size, number of parameters, and detection time but performed the best in recall, AP_{50}, and $AP_{50–95}$. The analysis above indicates that the YOLOv7-tiny + FasterNet model had the most satisfactory detection effect among the schemes. Figure 10 shows the detection of green walnut pictures under backlight. Among the three schemes, YOLOv7-tiny + FasterNet recognized the target with the highest confidence, and YOLOv7-tiny + LightMLP demonstrated moderate performance.

Table 1. Test results of three improvement schemes.

Algorithm	*P*/%	*R*/%	AP_{50}/%	$AP_{50–95}$/%	M_s/MB	P_a	*T*/ms
YOLOv7-tiny + FasterNet	95.6	94.7	96.2	76.6	10.1	4,933,548	13.3
YOLOv7-tiny + CARAFE	92.4	91.2	95.4	73.3	12.4	6,073,552	18.4
YOLOv7-tiny + LightMLP	97.0	90.3	96.0	76.2	12.7	6,215,276	17.9



(**a**) (**b**) (**c**)

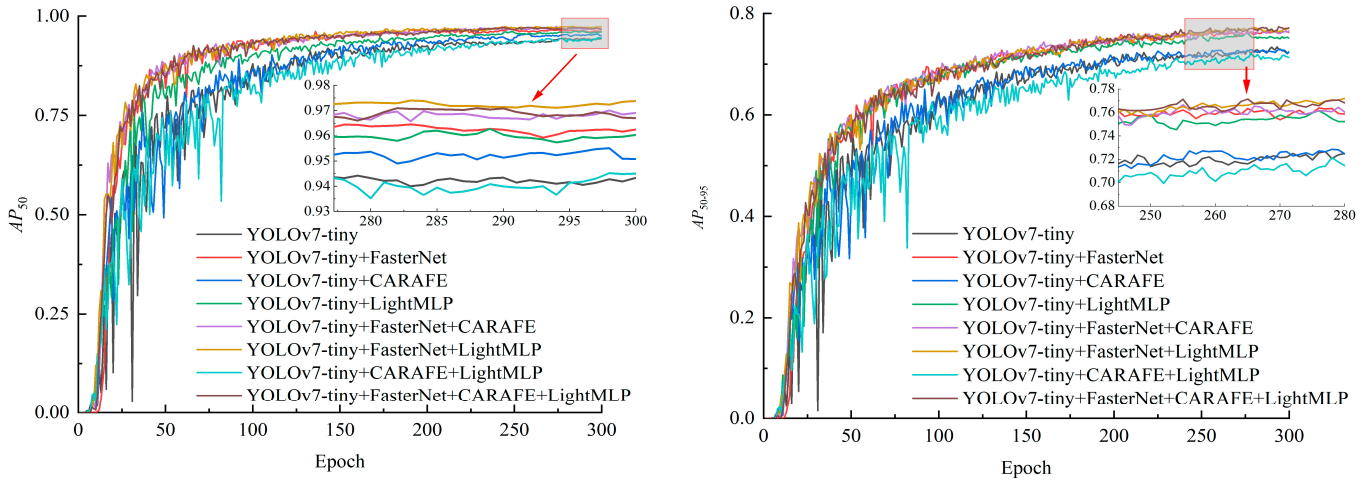
Figure 10. Single optimization detection comparison. (**a**) YOLOv7-tiny + FasterNet; (**b**) YOLOv7-tiny + CARAFE; (**c**) YOLOv7-tiny + LightMLP.

3.2. Ablation Test

3.2.1. Training Process of Model Improvement

The three improved strategies, namely, fusing the FasterNet module with the ELAN layer of the backbone network, introducing the CARAFE lightweight upsampling operator, and inserting the LightMLP module into the head structure, were used for an ablation test. Figure 11 shows the AP_{50} and $AP_{50–95}$ curves of the YOLOv7-tiny model and seven optimization strategies trained for 300 epochs. The figure can intuitively reveal the convergence of the average accuracy of each model. As shown in Figure 11a, the AP_{50} of YOLOv7-tiny that fuses FasterNet and the LightMLP module reached 75% before 50 training epochs, and the convergence effect was the best (the closest to 98% in the end). Although the final AP_{50} of the fusion of the CARAFE operator and LightMLP module exceeded that of the original network, the performance at the early and middle stages was not as good as that of the original network. The changes in AP_{50} in the training process showed that the training effect of the YOLOv7-tiny model after combining the CARAFE operator and LightMLP module was poor.

As indicated in Figure 11b, the $AP_{50–95}$ training effect of each model began to decline after 50 training epochs. The original model, the fused CARAFE operator, and the LightMLP module had poor training effects, and the final $AP_{50–95}$ values of the three modules were below 74%. By contrast, the final $AP_{50–95}$ values of the other models exceeded 75%.



(**a**) (**b**)

Figure 11. 8 models of AP_{50} and $AP_{50–95}$. (**a**) AP_{50}; (**b**) $AP_{50–95}$.

3.2.2. Test Results

The results of the tests conducted on the test set are shown in Table 2. YOLOv7-tiny, which was fused with the FasterNet module, outperformed the original network in all five performance indicators, and it improved the average accuracy while decreasing the model size and number of parameters. Compared with the original network, its detection time decreased by 2.3 ms, and its AP_{50} and $AP_{50–95}$ improved by 1.9% and 3.3%, respectively. Its model size was reduced to nearly 10 MB, and the number of parameters decreased by 17.9%. After YOLOv7-tiny was fused with the LightMLP module, its AP_{50} and $AP_{50–95}$ improved to 1.7% and 2.9%, respectively, relative to those of the original network, and the model size increased by 0.4 MB. The number of parameters increased by less than 3.5% in comparison with those of the original network, and the detection time increased by only 1.4 ms. The FasterNet and LightMLP modules resulted in considerable improvements in the YOLOv7-tiny model.

Table 2. Performance comparison between single optimization and combined optimization.

YOLOv7-Tiny	FasterNet	CARAFE	LightMLP	AP_{50}/%	$AP_{50–95}$/%	M_s/MB	P_a	T/ms
√ *				94.3	73.3	12.3	6,007,596	16.5
√	√			96.2	76.6	10.1	4,933,548	13.3
√		√		95.4	73.3	12.4	6,073,552	18.4
√			√	96.0	76.2	12.7	6,215,276	17.9
√	√	√		97.0	76.9	10.2	4,999,504	15.3
√	√		√	97.4	77.3	10.5	5,141,228	15.4
√		√	√	94.2	72.2	23.9	6,281,232	17.6
√	√	√	√	96.8	77.3	10.7	5,207,184	15.2

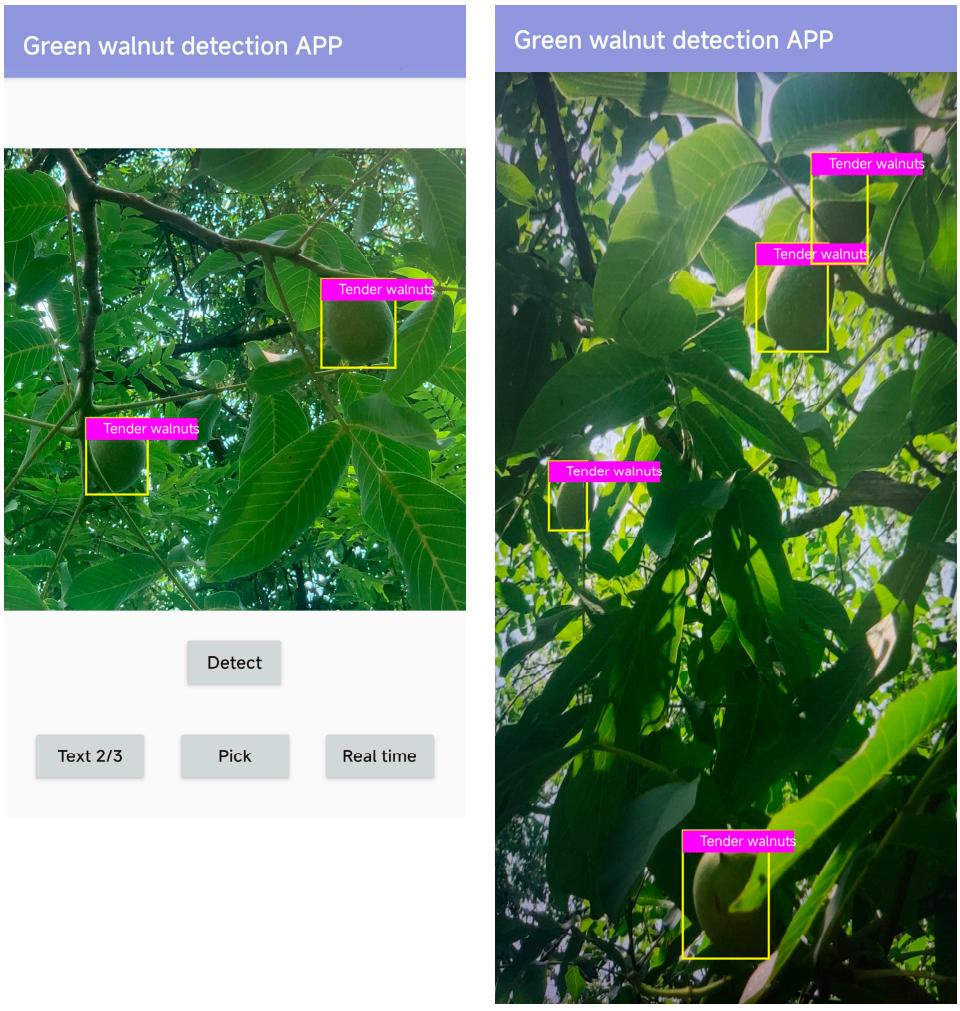
* The optimization was used.

In terms of the YOLOv7-tiny optimization using the combined strategy, ideal performance was achieved by fusing the FasterNet and LightMLP modules (FL-YOLOv7-tiny), which showed a large improvement compared with the original model in terms of AP_{50} and $AP_{50–95}$, with improvements of 3.1 percentage points for AP_{50} and 4 percentage points for $AP_{50–95}$. At the same time, the model size, number of parameters, and detection time decreased. The second-best performer was the fusion of the FasterNet and CARAFE operators, with a 2.7% improvement for AP_{50} and a 3.6% improvement for $AP_{50–95}$, in terms of reductions in model size, number of parameters, and detection time. The simultaneous introduction of the CARAFE operator and LightMLP module decreased the average accu-

racy of the model (i.e., by 1.1 percentage points for $AP_{50–95}$), indicating mutual exclusivity between the CARAFE operator and LightMLP module in the improvement of YOLOv7-tiny.

3.3. Android Deployment and Experimentation

The trained FL-YOLOv7-tiny model's pt file was converted into an onnx file through the export py script, and the onnx file was converted into an ncnn file and deployed to an Android phone by using Android Studio. The testing machine used Redmi K40 and Android 13 versions as an example, and the app could realize two modes of image detection and real-time detection. Figure 12a shows an example of image detection, and Figure 12b presents the main image of mobile phone real-time detection. Figure 12c shows the real-time detection of the left fine-tuning of the lens when shooting Figure 12b, and half of the green walnut in the upper right corner is framed. Figure 12d presents the real-time detection of the fine-tuning of the lens to the upper right when shooting Figure 12b. Two green walnuts with only a part of the fruit exposed and with a very dark color at the boundary of the real-time interface are boxed out.



(**a**) (**b**)

Figure 12. *Cont.*



(**c**) (**d**)

Figure 12. Example of Android terminal detection. (**a**) Image detection; (**b**) real-time detection of the main graph; (**c**) real-time detection of the left fine-tuning of the lens; (**d**) real-time detection of fine-tuned lens to the upper right.

After 15 photos were obtained in the field, the Android app was used for detection. The experimental results are shown in Table 3. The actual number of green walnuts in the photos was 83, and 77 of them were successfully detected, with a detection success rate of 92.8%. The detection success rate for the green walnuts with some fruits covered was 90.5%, and the detection success rate for the green walnuts without cover was 95.1%.

Table 3. Android APP detection experiment.

Experiment Number	Some of the Fruits Are Covered		The Fruits Are Not Covered		Total Number of Fruits	Total Number of Fruits Correctly Detected
	Sum	Number of Detections	Sum	Number of Detections		
1	3	3	2	2	5	5
2	1	1	0	0	1	1
3	1	1	3	3	4	4
4	0	0	6	6	6	6
5	3	2	0	0	3	2
6	5	4	2	2	7	6
7	0	0	8	8	8	8
8	6	6	5	4	11	10
9	3	3	2	2	5	5
10	0	0	1	1	1	1

Table 3. *Cont.*

Experiment Number	Some of the Fruits Are Covered		The Fruits Are Not Covered		Total Number of Fruits	Total Number of Fruits Correctly Detected
	Sum	Number of Detections	Sum	Number of Detections		
11	4	4	0	0	4	4
12	1	1	5	5	6	6
13	3	3	0	0	3	3
14	5	4	7	6	12	10
15	7	6	0	0	7	6

4. Discussion

4.1. Comparative Testing of CARAFE Performance under Different Parameters

Two parameter values, namely, encoder kernel size k_e and upsampled kernel size k_r (both of which are odd), need to be set to introduce the CARAFE operator into the YOLOv7-tiny structure. Wang et al. conducted experiments and empirically concluded that the training results improve when $k_r - k_e = 2$ [25]. On the basis of the test conditions and training verification, the maximum parameters that could be set were $k_e = 3$ and $k_r = 5$. The test results are shown in Table 4. The part with empty values in Schemes 4–7 represents the upsampling module that uses nearest neighbor interpolation, and Scheme 7 is the performance of the original YOLOv7-tiny.

Table 4. The performance comparison of CARAFE under different parameters is introduced.

Scheme Sequence Number	Upsample1-k_e	Upsample1-k_r	Upsample2-k_e	Upsample2-k_r	AP_{50}/%	AP_{50-95}/%	M_s/MB	P_a	T/ms
1	1	3	1	3	94.5	72.2	12.3	6,024,692	20.4
2	3	5	1	3	94.6	73.1	12.5	6,080,052	20.7
3	1	3	3	5	94.5	73.1	12.5	6,080,052	21.5
4	1	3	-	-	94.4	72.7	12.3	6,018,192	17.5
5	-	-	1	3	94.4	72.0	12.3	6,014,096	19.0
6	3	5	-	-	95.4	73.3	12.4	6,073,552	18.4
7	-	-	-	-	94.3	73.3	12.3	6,007,596	16.5

As indicated in Table 4, in Schemes 1–5, the AP_{50-95} of the models that replaced the CARAFE operator decreased at different degrees. Compared with the AP_{50-95} of the original model, the AP_{50-95} of Scheme 5 had a larger amount of reduction (up to 1.3%), and Schemes 2 and 3 had a smaller amount of reduction (only 0.2%). After the CARAFE operator was replaced, the AP_{50} of each model improved, and the amount of improvement was distributed from 0.1% to 1.1%. The number of model parameters of Scheme 6 increased by 1.1% relative to that of the original model; its detection time increased by only 1.9 ms, and its AP_{50} reached 95.4%. Only in this scheme did AP_{50-95} not decrease, which verifies the effectiveness of the YOLOv7-tiny model after the introduction of the CARAFE operator.

4.2. Performance Comparison of Different Algorithms

The same hyperparameters were set, and the currently popular YOLO family of algorithms was trained and compared on a test set. The performance comparison of the different algorithms is shown in Table 5. The YOLOv7-tiny model, which integrates FasterNet and LightMLP modules, had the highest AP_{50} among the seven models, and its five indicators improved considerably in comparison with those of the unoptimized YOLOv7-tiny model. At the same time, the AP_{50-95} value of FL-YOLOv7-tiny was close to that of the YOLOv5s model, but its number of parameters, model size, and detection time decreased by 26.7%, 27.6%, and 27.7%, respectively, in comparison with those of the YOLOv5s model. The advantage of YOLOv5n was that it had the smallest model size, parameter number, and detection time among the models, but its AP_{50} and AP_{50-95} were lower than those of YOLOv5s. Among the compared models, YOLOv3-tiny had the largest number of parameters, and its AP_{50} and AP_{50-95} were lower than those of FL-YOLOv7-tiny.

The AP_{50} of the YOLOv6-3.0s model was lower than that of FL-YOLOv7-tiny in this study, but the performance in the four other indicators was good in the test set. Although the $AP_{50–95}$ of YOLOv8s was higher than that of FL-YOLOv7-tiny, YOLOv8s had the largest model size among the models, its number of parameters was more than twice that of the optimized model in this study, and its detection time was the longest. Compared with YOLOv8s, YOLOv8n had a smaller model size and parameter number and demonstrated accelerated detection; its AP_{50} and $AP_{50–95}$ were reduced.

Table 5. Performance comparison of different algorithms.

Algorithm	AP_{50}/%	$AP_{50–95}$/%	M_s/MB	P_a	T/ms
YOLOv3-tiny	96.7	76.5	22.4	12,128,178	14.5
YOLOv5n	95.7	75.3	3.9	1,760,518	11.5
YOLOv5s	96.4	77.4	14.5	7,012,822	21.3
YOLOv6-3.0s	97.1	79.3	8.7	4,233,843	13.3
YOLOv8n	96.6	78.4	6.3	3,005,843	13.5
YOLOv8s	97.1	80.1	22.5	11,125,971	25.9
YOLOv7-tiny	94.3	73.3	12.3	6,007,596	16.5
FL-YOLOv7-tiny	97.4	77.3	10.5	5,141,228	15.4

In terms of the performance in the five indicators, FL-YOLOv7-tiny exhibited certain advantages not only in the performance improvement of the original YOLOv7-tiny network but also in the performance comparison of various YOLO series models. FL-YOLOv7-tiny demonstrated superiority after YOLOv7-tiny model optimization and is thus suitable for deployment in smart terminal devices.

Figure 13 shows the test set detection results of eight models. The eight models were YOLOv3-tiny, YOLOv5n, YOLOv5s, YOLOv6-3.0s, YOLOv8n, YOLOv8s, YOLOv7-tiny, and FL-YOLOv7-tiny. The original YOLOv7-tiny suffered from misdetection, that is, it recognized a part of the leafy background in the far lower-right corner as a green-barked walnut (Figure 13h). The seven other models did not produce false detections. The YOLOv7-tiny model that fused FasterNet and LightMLP modules had the most outstanding performance among the eight models, and its confidence level of monomer identification exceeded 90%, with 97%, 96%, 92%, and 95% confidence from top to bottom.

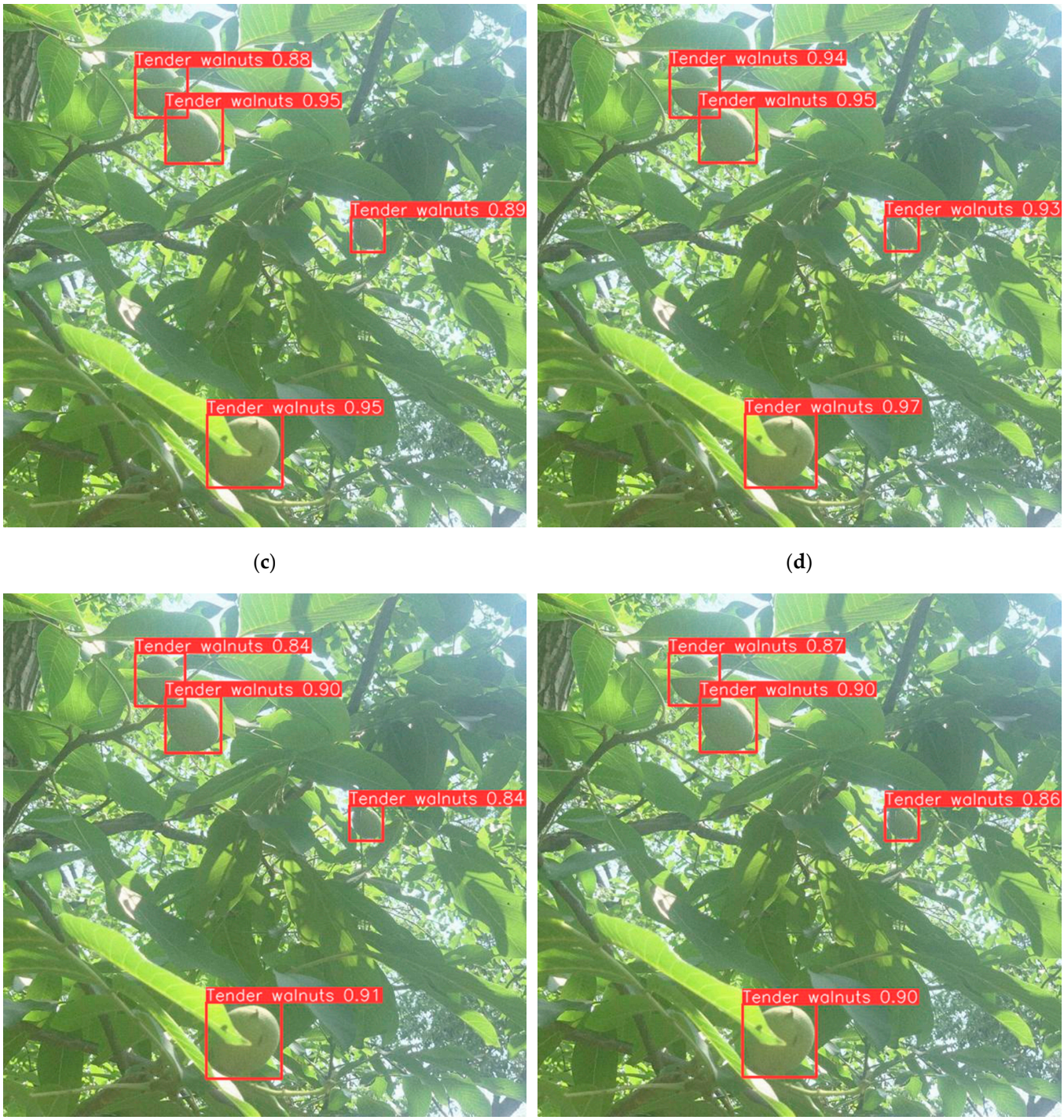


(**a**)



(**b**)

Figure 13. *Cont.*



(**c**) (**d**)

(**e**) (**f**)

Figure 13. *Cont*.



(**g**)



(**h**)



(**i**)

Figure 13. The detection effect of different algorithms on the images in the test set. (**a**) One of the detected images in the test set; (**b**) YOLOv3-tiny; (**c**) YOLOv5n; (**d**) YOLOv5s; (**e**) YOLOv6-3.0s; (**f**) YOLOv8n; (**g**) YOLOv8s; (**h**) YOLOv7-tiny; (**i**) FL-YOLOv7-tiny.

4.3. Potential Application and Future Study

The object of this study is green walnuts, but the proposed optimization model of target detection can be used for the detection not only of green walnuts but also of other crops with a similar leaf color. When the model is used for target detection of other crops, such as green apples, oranges, or peppers, the only requirement is that the new datasets must be obtained, labeled, and trained for the trained FL-YOLOv7-tiny.

Fan et al. performed green walnut detection and used the Adam optimizer in a detection model that incorporates VGG16 into the Faster R-CNN model; they found that the combined model exhibits enhanced performance [12]. The optimized Faster R-CNN model has a detection time of 0.5 s, but the researchers did not provide results on the model size and number of parameters. This combined model cannot easily meet the current requirements of real-time detection. Hao et al. introduced the MobileNet-v3 backbone network into the YOLOv3 model, but although the new model's detection speed is improved compared with that of the Faster R-CNN model, the network size is close to 100 MB and occupies a large space, which is not conducive to model nesting [13].

The main reason for choosing YOLOv7-tiny in this study is that the model has a small size and high detection efficiency and is suitable for embedding in mobile terminals and hardware devices. The size of the optimized FL-YOLOv7-tiny is only 10.5 MB, the number of references is 5,141,228, and the detection time is only 15.4 ms.

In today's rapidly developing scientific and technological fields, the cross fertilization of different technologies has become a trend. Miniaturized near-infrared spectroscopy has been utilized to analyze the quality of various crops and has produced valuable results [38]. In future research, the proposed target detection model can be combined with miniaturized near-infrared spectroscopy for applications in the whole process of detecting fruits first and then analyzing quality later.

The dataset in this study was obtained in sunny weather. In future studies, further consideration should be given to the weather factor and data volume during dataset acquisition.

5. Conclusions

In this study, three improvement strategies, namely, the introduction of a FasterNet module that fused PConv with the ReLU activation function, the use of CARAFE in combination with nearest-neighbor interpolation upsampling, and the insertion of the LightMLP module into the head structure, were adopted to explore the changes in model performance after single and combined optimization through ablation tests. The results showed that the YOLOv7-tiny model that integrated FasterNet and LightMLP modules was the most effective among all the models. Its AP_{50} and $AP_{50–95}$ improved to 97.4% and 77.3%, respectively, and its model size was 10.5 MB. Its number of parameters decreased to 5,141,228, and its detection time was only 15.4 ms. In conclusion, compared with the original model, the FL-YOLOv7-tiny model shows considerable improvement and can be deployed to terminal devices for intelligent real-time detection in walnut orchards.

Author Contributions: Conceptualization, X.F. and F.Z. (Fengzi Zhang); methodology, X.F. and F.Z. (Fengzi Zhang); software, X.F. and W.P.; validation, W.P. and Y.Z.; formal analysis, Y.Z.; resources, J.W.; data curation, X.F.; writing—original draft preparation, X.F.; writing—review and editing, J.W.; visualization, F.Z. (Fu Zhao); supervision, F.Z. (Fu Zhao); project administration, J.W.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (grant number 11802167), and the Key Research Project of Shanxi Province (grant number 202102020101012).

Data Availability Statement: Data is available on request due to privacy.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Deng, J.; Pan, Q.; Liu, A. Study on the characteristics of main nutrients of 16 domestic walnut varieties. *Food Ferment. Sci. Technol.* **2023**, *3*, 111–115.
2. Jie, M.; Chen, B.; Wu, X.; Wang, X. Research status and suggestions for adjustment of fresh walnut preservation. *Gansu Agric. Sci. Technol.* **2020**, *4*, 68–71.
3. Fan, X.; Xu, Y.; Zhou, J.; Liu, X.; Tang, J. Identification and localization of walnut based on improved Faster R-CNN. *J. Yanshan Univ.* **2021**, *6*, 544–551.

4. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
5. Girshick, R. Fast r-cnn. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
6. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
7. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part I 14. Springer International Publishing: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
8. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
9. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
10. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A single-stage object detection framework for industrial applications. *arXiv* **2022**, arXiv:2209.02976.
11. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 7464–7475.
12. Fan, X.; Xu, Y.; Zhou, J.; Liu, X.; Tang, J.; Wei, Y. Green Walnut Detection Method Based on Improved Convolutional Neural Network. *Trans. Chin. Soc. Agric. Mach.* **2021**, *9*, 149–155.
13. Hao, J.; Bing, Z.; Yang, S.; Yang, J.; Sun, L. Detection of walnut by improved YOLOv3 algorithm. *Trans. Chin. Soc. Agric. Eng.* **2022**, *14*, 183–190.
14. Tian, Y.; Yang, G.; Wang, Z.; Wang, H.; Li, E.; Liang, Z. Apple detection during different growth stages in orchards using the improved YOLO-V3 model. *Comput. Electron. Agric.* **2019**, *157*, 417–426. [CrossRef]
15. Song, H.; Ma, B.; Shang, Y.; Wen, Y.; Zhang, S. Detection of young apple fruits based on YOLO v7-ECA model. *Trans. Chin. Soc. Agric. Mach.* **2023**, *6*, 233–242.
16. Jiang, M. Research on Detection Method of Young Apple Fruit in Near Scenery by Integrating Convolutional Neural Network and Visual Attention Mechanism. M.S. Thesis, Northwest A&F University, Yangling, China, 2022.
17. Chen, G. Research on Target Recognition and Picking Point Determination Technology of Green Pepper in Greenhouse Environment. M.S. Thesis, Jiangsu University, Suzhou, China, 2020.
18. Bargoti, S.; Underwood, J. Deep fruit detection in orchards. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 3626–3633.
19. Hu, G.; Zhou, J.; Chen, C.; Li, C.; Sun, L.; Chen, Y.; Zhang, S.; Chen, J. Detection method of apple in orchard environment by combining lightweight network and attention mechanism. *Trans. Chin. Soc. Agric. Eng.* **2022**, *19*, 131–142.
20. Zhang, J.; Chen, W.; Wei, Q.; Guo, B. Based on Des-YOLO v4 under the complex environment of apple detection method. *J. Agric. Mech. Res.* **2023**, *5*, 20–25.
21. Sun, J.; Qian, L.; Zhu, W.; Zhou, X.; Dai, C. Detection of apples in complex orchard environment based on improved RetinaNet. *Trans. Chin. Soc. Agric. Eng.* **2022**, *15*, 314–322.
22. Zhu, Z.; Shan, J.; Yu, X.; Kong, D.; Wang, Q.; Xie, X. Target detection technology of tomato picking robot based on YOLOv5s. *Transducer Microsyst. Technol.* **2023**, *6*, 129–132.
23. Liu, J.; He, J.; Chen, H.; Wang, X.; Zhai, H. Tomato string detection model based on improved YOLOv4 and ICNet. *Trans. Chin. Soc. Agric. Mach.* **2023**, *10*, 216–224+254.
24. Zhao, H.; Li, W.; Yang, Z. Detection of tomato target and grasping position in rain weather based on improved YOLOv4. *Jiangsu Agric. Sci.* **2023**, *1*, 202–210.
25. Wang, J.; Zhou, J.; Zhang, Y.; Hu, H. Multi-pose dragon fruit detection system of picking robot based on optimal YOLOv7 model. *Trans. Chin. Soc. Agric. Eng.* **2019**, *8*, 276–283.
26. Zhou, J.; Wang, J.; Zhang, Y.; Hu, H. Pitaya rapid detection method based on GCAM-YOLOv5. *J. For. Eng.* **2023**, *3*, 141–149.
27. Ma, R.; He, H.; Chen, Y.; Lai, Y.; Jiao, R.; Tang, H. Maturity identification method of dragon fruit based on improved YOLOv5. *J. Shenyang Agric. Univ.* **2023**, *2*, 196–206.
28. Xiong, J.; Huo, Z.; Huang, Q.; Chen, H.; Yang, Z.; Huang, Y.; Su, Y. Detection method of citrus in nighttime environment combined with active light source and improved YOLOv5s model. *J. South China Agric. Univ.* **2024**, *1*, 97–107.
29. Gao, X.; Wei, S.; Wen, Z.; Yu, T. Improved citrus detection method by YOLOv5 lightweight network. *Comput. Eng. Appl.* **2023**, *11*, 212–221. [CrossRef]
30. Huang, T.; Huang, H.; Li, Z.; Lv, S.; Xue, X.; Dai, Q.; Wen, W. Citrus fruit recognition method based on the improved model YOLOv5. *J. Huazhong Agric. Univ.* **2022**, *04*, 170–177.
31. Chen, J.; Kao, S.H.; He, H.; Zhuo, W.; Wen, S.; Lee, C.H.; Chan, S.H.G. Run, don't walk: Chasing higher FLOPS for faster neural networks. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 20–22 June 2023; pp. 12021–12031.

32. Wang, J.; Chen, K.; Xu, R.; Liu, Z.; Loy, C.C.; Lin, D. Carafe: Content-aware reassembly of features. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3007–3016.
33. Liu, M.; Wan, L.; Wang, B.; Wang, T. SE-YOLOv4: Shuffle expansion YOLOv4 for pedestrian detection based on PixelShuffle. *Appl. Intell.* **2023**, *53*, 18171–18188. [CrossRef]
34. Quan, Y.; Zhang, D.; Zhang, L.; Tang, J. Centralized feature pyramid for object detection. In *IEEE Transactions on Image Processing (TIP)*; IEEE: Piscataway, NJ, USA, 2023; Volume 43, pp. 4341–4354.
35. Yu, W.; Luo, M.; Zhou, P.; Si, C.; Zhou, Y.; Wang, X.; Feng, J.; Yan, S. Metaformer is actually what you need for vision. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 10819–10829.
36. Larsson, G.; Maire, M.; Shakhnarovich, G. Fractalnet: Ultra-deep neural networks without residuals. *arXiv* **2016**, arXiv:1605.07648.
37. Tolstikhin, I.O.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yung, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; et al. Mlp-mixer: An all-mlp architecture for vision. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 24261–24272.
38. Beć, K.B.; Grabska, J.; Huck, C.W. Miniaturized NIR spectroscopy in food analysis and quality control: Promises, challenges, and perspectives. *Foods* **2022**, *11*, 1465. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.